Run | Debug | Stop | Share | Save | { } Beautify | Language Python 3

main.py

```
1  def convert_temperature(celsius):
2      kelvin = celsius + 273.15
3      fahrenheit = celsius * 1.8 + 32
4      return [round(kelvin, 5), round(fahrenheit, 5)]
5
6
7  celsius = 36.50
8  result = convert_temperature(celsius)
9  print(result)
10
11
```

input

```
[309.65, 97.7]


...Program finished with exit code 0
Press ENTER to exit console.
```

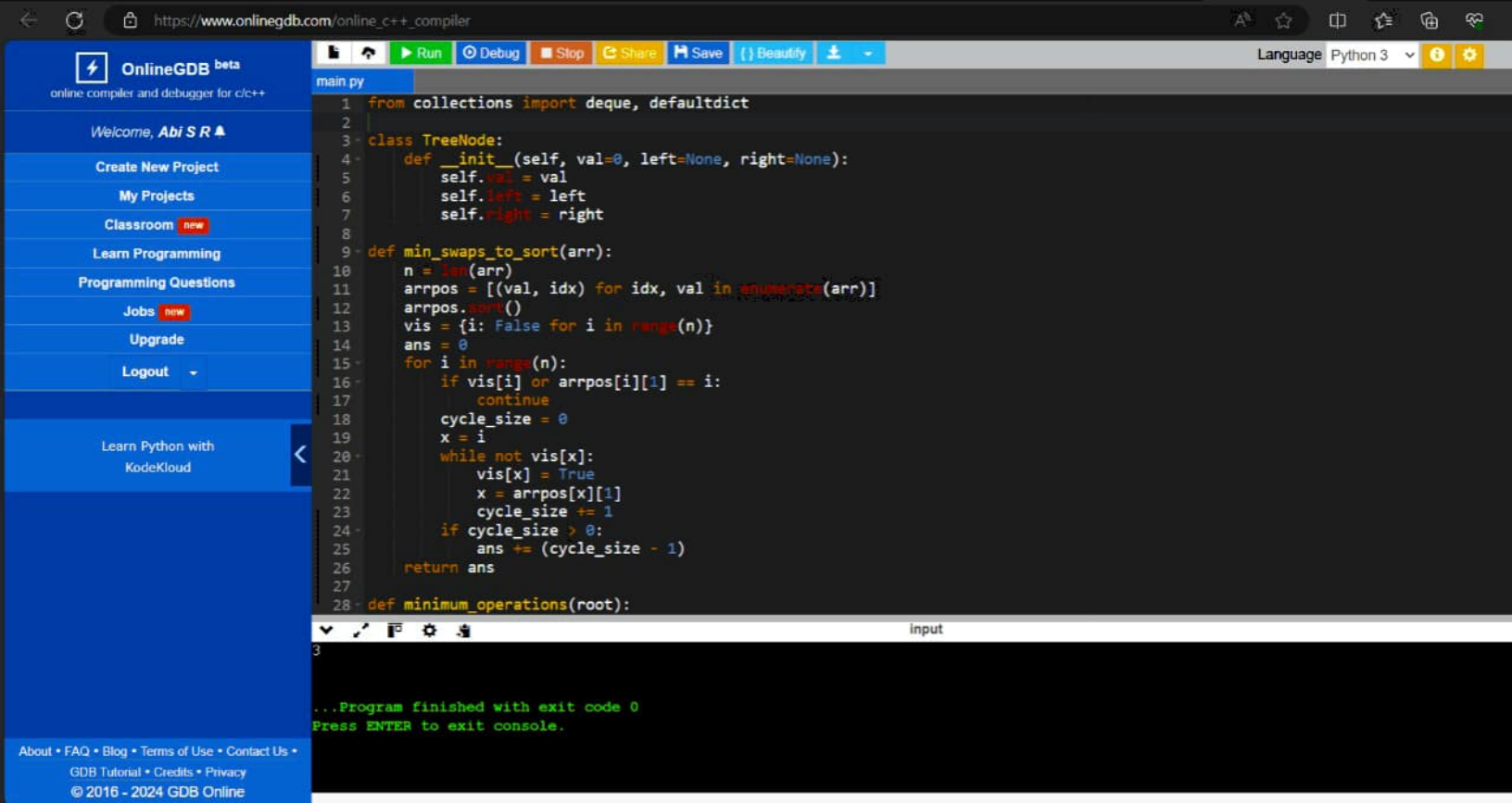
Run | Debug | Stop | Share | Save | { } Beautify | Language Python 3

main.py

```python
from math import gcd

def count_subarrays_with_lcm_equal_to_k(nums, k):
    def lcm(a, b):
        return abs(a*b) // gcd(a, b)

    def count_divisible(arr, k):
        count = 0
        for i in range(len(arr)):
            mul = 1
            for j in range(i, len(arr)):
                mul = lcm(mul, arr[j])
                if mul == k:
                    count += 1
        return count

    return count_divisible(nums, k)

nums = [3, 6, 2, 7, 1]
k = 6
print(count_subarrays_with_lcm_equal_to_k(nums, k))  # Output: 4
```

input

```
4


...Program finished with exit code 0
Press ENTER to exit console.
```


https://www.onlinegdb.com/online_c++_compiler
OnlineGDB beta
online compiler and debugger for c/c++
Welcome, Abi S R
Create New Project
My Projects
Classroom new
Learn Programming
Programming Questions
Jobs new
Upgrade
Logout
Learn Python with
KodeKloud
Run
Debug
Stop
Share
Save
{} Beautify
Language
Python 3
main.py
from collections import deque, defaultdict
class TreeNode:
def __init__(self, val=0, left=None, right=None):
self.val = val
self.left = left
self.right = right
def min_swaps_to_sort(arr):
n = len(arr)
arrpos = [(val, idx) for idx, val in enumerate(arr)]
arrpos.sort()
vis = {i: False for i in range(n)}
ans = 0
for i in range(n):
if vis[i] or arrpos[i][1] == i:
continue
cycle_size = 0
x = i
while not vis[x]:
vis[x] = True
x = arrpos[x][1]
cycle_size += 1
if cycle_size > 0:
ans += (cycle_size - 1)
return ans
def minimum_operations(root):
Input
3
...Program finished with exit code 0
Press ENTER to exit console.
About • FAQ • Blog • Terms of Use • Contact Us •
GDB Tutorial • Credits • Privacy
© 2016 - 2024 GDB Online

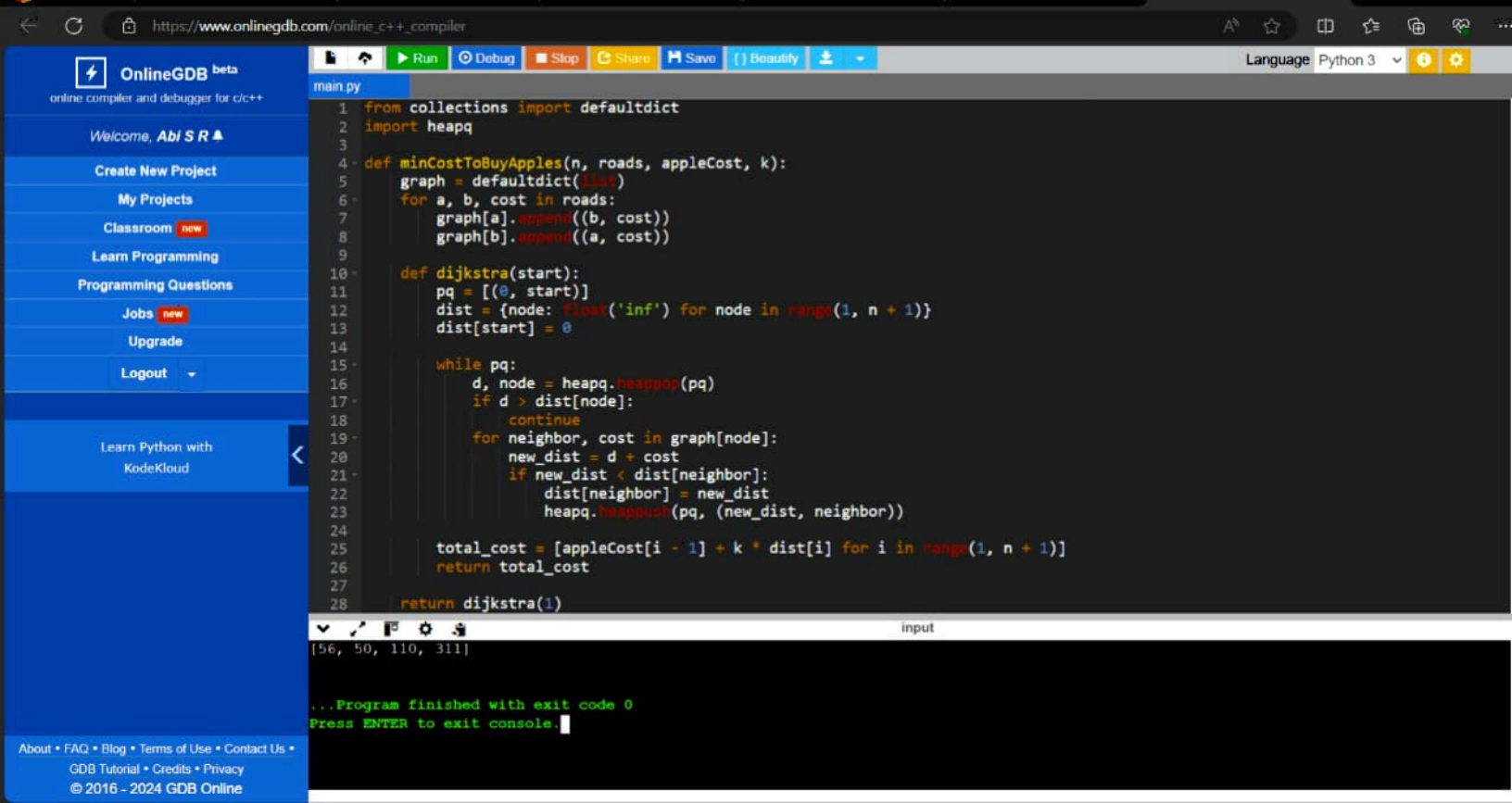
Run | Debug | Stop | Share | Save | { } Beautify | Language Python 3

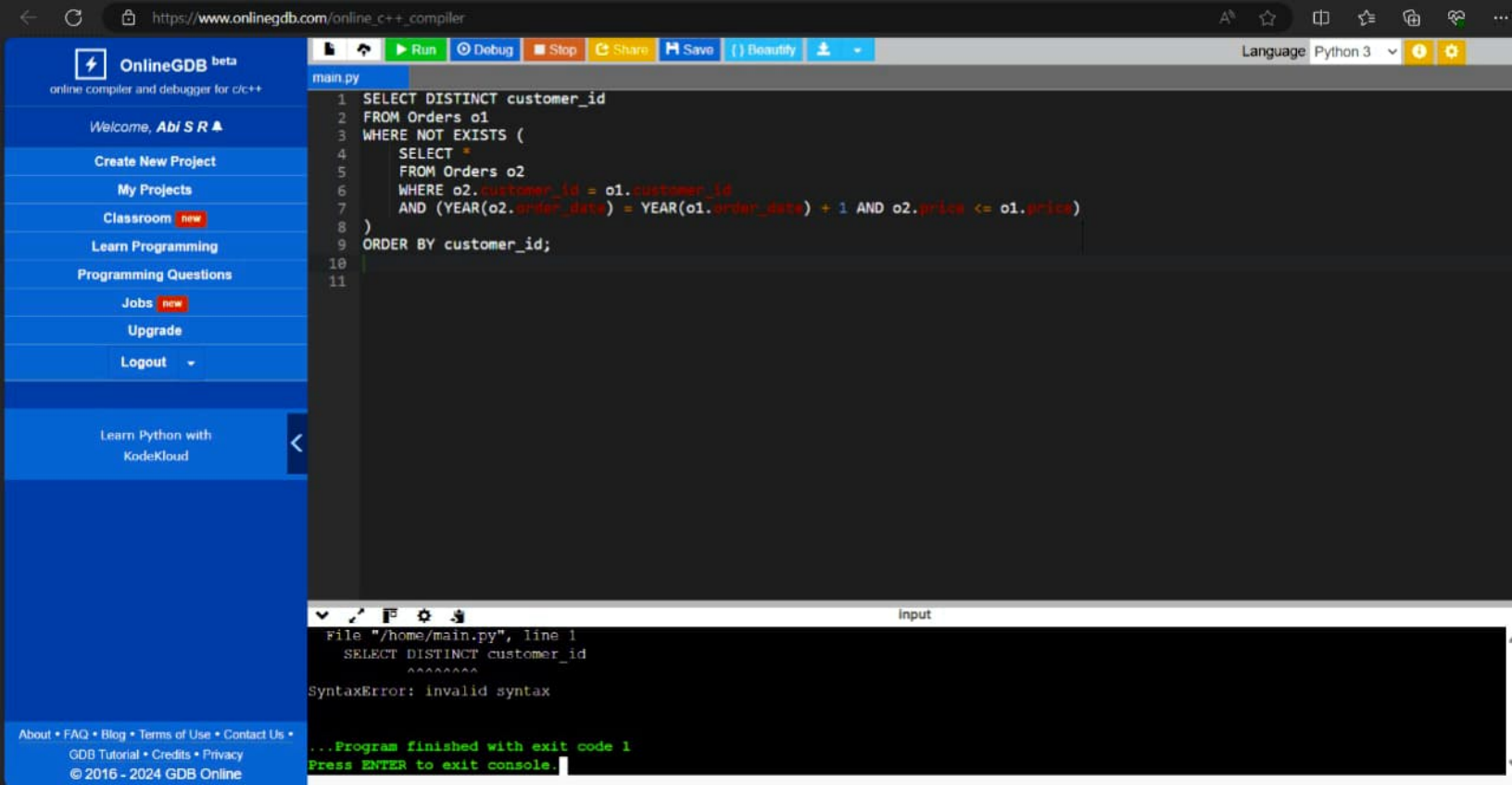
main.py

```python
def max_num_of_palindromes(s, k):
    def is_palindrome(sub):
        return sub == sub[::-1]

    def backtrack(start, path):
        nonlocal max_count
        if start == len(s):
            max_count = max(max_count, len(path))
            return

        for end in range(start + k, len(s) + 1):
            sub = s[start:end]
            if is_palindrome(sub):
                path.append(sub)
                backtrack(end, path)
                path.pop()

    max_count = 0
    backtrack(0, [])
    return max_count


s = "abaccdbbd"
k = 3
print(max_num_of_palindromes(s, k))
```

input

```
0


...Program finished with exit code 0
Press ENTER to exit console.
```


https://www.onlinegdb.com/online_c++_compiler
OnlineGDB beta
online compiler and debugger for c/c++
Welcome, Abi S R
Create New Project
My Projects
Classroom new
Learn Programming
Programming Questions
Jobs new
Upgrade
Logout
Learn Python with
KodeKloud
Run
Debug
Stop
Share
Save
{ } Beautify
Language
Python 3
main.py
from collections import defaultdict
import heapq
def minCostToBuyApples(n, roads, appleCost, k):
graph = defaultdict(list)
for a, b, cost in roads:
graph[a].append((b, cost))
graph[b].append((a, cost))
def dijkstra(start):
pq = [(0, start)]
dist = {node: float('inf') for node in range(1, n + 1)}
dist[start] = 0
while pq:
d, node = heapq.heappop(pq)
if d > dist[node]:
continue
for neighbor, cost in graph[node]:
new_dist = d + cost
if new_dist < dist[neighbor]:
dist[neighbor] = new_dist
heapq.heappush(pq, (new_dist, neighbor))
total_cost = [appleCost[i - 1] + k * dist[i] for i in range(1, n + 1)]
return total_cost
return dijkstra(1)
input
[56, 50, 110, 311]
...Program finished with exit code 0
Press ENTER to exit console.
About • FAQ • Blog • Terms of Use • Contact Us •
GDB Tutorial • Credits • Privacy
© 2016 - 2024 GDB Online


https://www.onlinegdb.com/online_c++_compiler
OnlineGDB beta
online compiler and debugger for c/c++
Welcome, Abi S R
Create New Project
My Projects
Classroom new
Learn Programming
Programming Questions
Jobs new
Upgrade
Logout
Learn Python with KodeKloud
Run
Debug
Stop
Share
Save
{ } Beautify
Language Python 3
main.py
1 SELECT DISTINCT customer_id
2 FROM Orders o1
3 WHERE NOT EXISTS (
4 SELECT *
5 FROM Orders o2
6 WHERE o2.customer_id = o1.customer_id
7 AND (YEAR(o2.order_date) = YEAR(o1.order_date) + 1 AND o2.price <= o1.price)
8)
9 ORDER BY customer_id;
10
11
Input
File "/home/main.py", line 1
SELECT DISTINCT customer_id
^^^^^^^^
SyntaxError: invalid syntax
...Program finished with exit code 1
Press ENTER to exit console.
About • FAQ • Blog • Terms of Use • Contact Us •
GDB Tutorial • Credits • Privacy
© 2016 - 2024 GDB Online

Run | Debug | Stop | Share | Save | { } Beautify | Language Python 3

main.py

```python
def count_good_triplets(nums):
    count = 0
    n = len(nums)
    for i in range(n):
        for j in range(i+1, n):
            for k in range(j+1, n):
                if nums[i] != nums[j] and nums[i] != nums[k] and nums[j] != nums[k]:
                    count += 1
    return count

nums = [4, 4, 2, 4, 3]
print(count_good_triplets(nums))
```

input

```
3


...Program finished with exit code 0
Press ENTER to exit console.
```

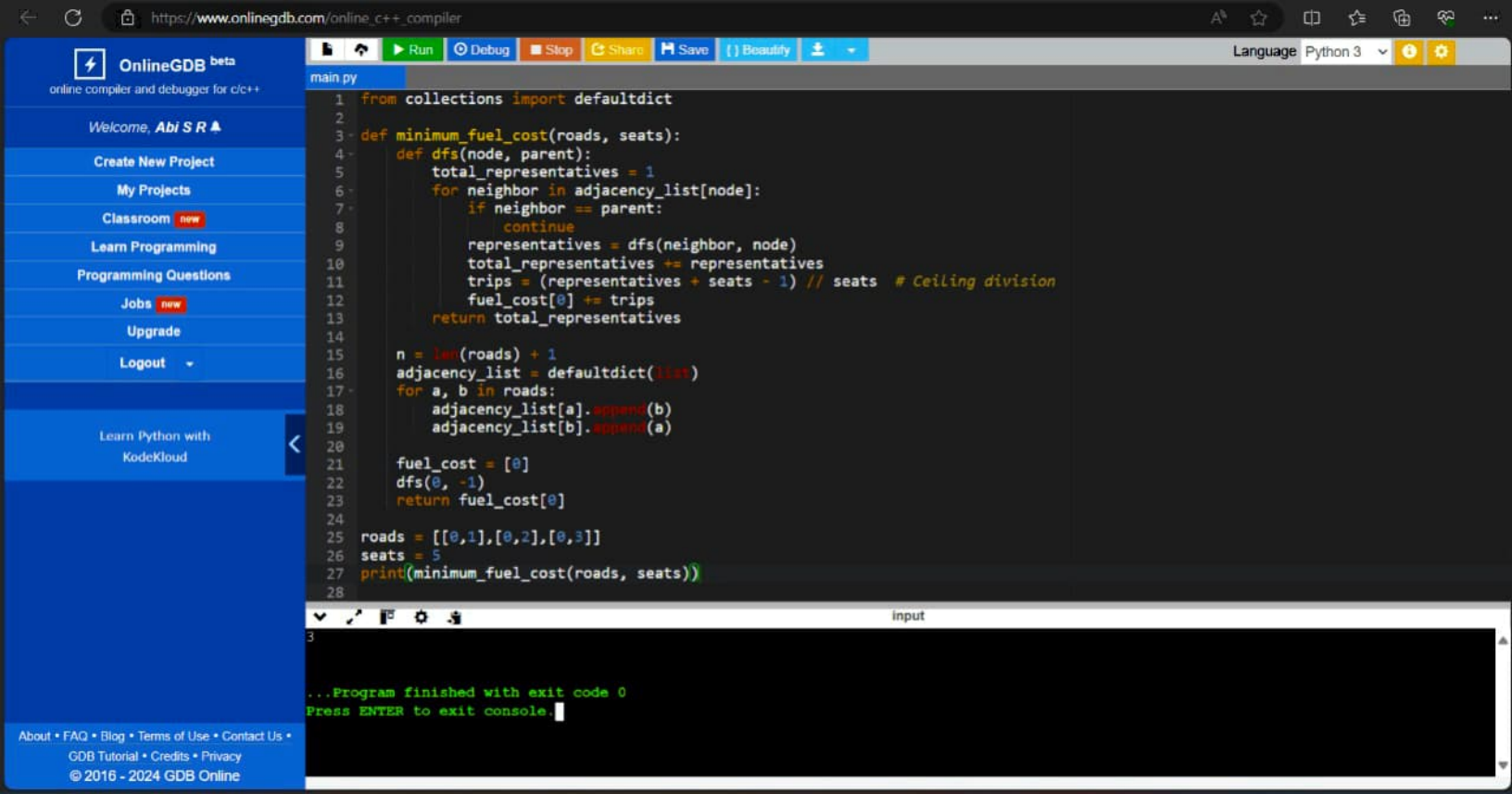
Run | Debug | Stop | Share | Save | { } Beautify | Language Python 3

main.py

```python
class TreeNode:
    def __init__(self, val=0, left=None, right=None):
        self.val = val
        self.left = left
        self.right = right

def closestNodesQueries(root, queries):
    def findClosest(node, target, closest):
        if not node:
            return closest
        if node.val <= target:
            closest[0] = max(closest[0], node.val)
            findClosest(node.right, target, closest)
        else:
            closest[1] = min(closest[1], node.val)
            findClosest(node.left, target, closest)

    def findClosestNodes(root, query):
        closest = [-1, -1]
        findClosest(root, query, closest)
        return closest

    result = []
    for q in queries:
        result.append(findClosestNodes(root, q))

    return result

```

input

```
[[2, -1], [4, -1], [15, -1]]


...Program finished with exit code 0
Press ENTER to exit console.
```


https://www.onlinegdb.com/online_c++_compiler
OnlineGDB beta
online compiler and debugger for c/c++
Welcome, Abi S R
Create New Project
My Projects
Classroom new
Learn Programming
Programming Questions
Jobs new
Upgrade
Logout
Learn Python with KodeKloud
Run
Debug
Stop
Share
Save
{ } Beautify
Language Python 3
main.py
from collections import defaultdict
def minimum_fuel_cost(roads, seats):
def dfs(node, parent):
total_representatives = 1
for neighbor in adjacency_list[node]:
if neighbor == parent:
continue
representatives = dfs(neighbor, node)
total_representatives += representatives
trips = (representatives + seats - 1) // seats # Ceiling division
fuel_cost[0] += trips
return total_representatives
n = len(roads) + 1
adjacency_list = defaultdict(list)
for a, b in roads:
adjacency_list[a].append(b)
adjacency_list[b].append(a)
fuel_cost = [0]
dfs(0, -1)
return fuel_cost[0]
roads = [[0,1],[0,2],[0,3]]
seats = 5
print(minimum_fuel_cost(roads, seats))
input
3
...Program finished with exit code 0
Press ENTER to exit console.
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2024 GDB Online

Run | Debug | Stop | Share | Save | { } Beautify | Language Python 3

main.py

```python
def count_beautiful_partitions(s, k, minLength):
    MOD = 10**9 + 7
    primes = {'2', '3', '5', '7'}

    def is_prime_digit(digit):
        return digit in primes

    def count_partitions(s, k, minLength, start, end):
        if k == 0:
            return 1 if start == end else 0
        if end - start < k * minLength:
            return 0

        count = 0
        for i in range(start + minLength, end):
            if is_prime_digit(s[i]):
                count += count_partitions(s, k - 1, minLength, i + 1, end)

        return count % MOD

    return count_partitions(s, k, minLength, 0, len(s))

s = "23542185131"
k = 3
minLength = 2
print(count_beautiful_partitions(s, k, minLength))
```

input

```
0


...Program finished with exit code 0
Press ENTER to exit console.
```